Experimental Comparisons of Multi-class Classifiers

Lin Li
Institute of Intelligent Computing and Information Technology, Chengdu Normal University
No.99, East Haike Road Wenjiang District, Chengdu, China
E-mail: lilin200909@gmail.com

Lin Li, Yue Wu and Mao Ye
School. of Computer Science and Engineering, University of Electronic Science and Technology of China
No.2006, Xiyuan Ave, West Hi-Tech Zone, Chengdu, China

Keywords: multi-class classifier, tree classifier, boost classifier, support vector machines, combining boost classifier

Received: June 18, 2014

The multi-class classification algorithms are widely used by many areas such as machine learning and computer vision domains. Nowadays, many literatures described multi-class algorithms, however there are few literature that introduced them with thorough theoretical analysis and experimental comparisons. This paper discusses the principles, important parameters, application domain, runtime performance, accuracy, and etc. of twelve multi-class algorithms: decision tree, random forests, extremely randomized trees, multi-class adaboost classifier, stochastic gradient boosting, linear and nonlinear support vector machines, K nearest neighbors, multi-class logistic classifier, multi-layer perceptron, naive Bayesian classifier and conditional random fields classifier. The experiment tested on five data sets: SPECTF heart data set, Ionosphere radar data set, spam junk mail filter data set, optdigits handwriting data set and scene 15 image classification data set. Our major contribution is that we study the relationships between each classifier and impact of each parameters to classification results. The experiment shows that gradient boosted trees, nonlinear support vector machine, K nearest neighbor reach high performance under the circumstance of binary classification and minor data capacity; Under the condition of high dimension, multi-class and big data, however, gradient boosted trees, linear support vector machine, multi-class logistic classifier get good results. At last, the paper addresses the development and future of multi-class classifier algorithms.

Povzetek: V prispevku je podan pregled klasifikatorjev z več razredi.

1 Introduction

Multi-classification problem [1] is that of supposing a set of training data $(x_1,c_1),...,(x_n,c_n)$, where $x \in R^p$ are finite set of input features, $c_i \in \{1,2,...,K\}$ are class numbers of output variables. The purpose of multi-classification task is to find a classifying rules based on the training sample, then given a new features, outputting a classifying category. Today multi-classifier algorithms are applied to a variety of application areas such as: radar signal classification, character recognition, remote sensing, medical diagnostics, expert systems, voice recognition domains and etc.

Multiple classifiers has a long history. Selfridge et al. [2] first propose a multi-classification system based on ‘winners get all’ solution which chooses the optimal solution as a multi-classifier output. Kanal and Minsky [3, 4] play an important role in multi-classifiers development. They claim that any classifying algorithm does not solve all problems. We need to design specific classifying algorithm for different problems. Multilayer perceptron [5] is an artificial neural network model that can resolve this kind of nonlinear data. Decision tree is an ancient non-parametric classification algorithm that classifies the samples according to the classifying rules. Leo Breiman[6] proposes random forest as a good solution to the scalability issues of single decision tree. Adboost algorithm is proposed by Yoav Freund and Robert Schapire[34] is a meta-classification algorithm which can be combined with other classification algorithms to enhance its performance. Multi-class logistic classifier proposed by Jerome Friedman et al. [7] is another important improvement of enhancing the basic boosting algorithm. K nearest neighbor [5] classifies samples based on the adjacent spatial relationships of features. In 1980s with the rise of data fusion and learning model in statistics and management science, Bayesian expert [8-10] system is widely used. Since the 1990s, Vapnik proposes support vector machines, transforming the feature from low dimensional space into high dimensional space, which is a better ways to classify the features. Nonlinear support vector machine gets a great success, however it is not ideal in some cases i.e. the original features are already high dimensional space, so people propose a linear support vector machine[11] for these cases. Because of the complexity of the data, a single classifier is often difficult to obtain good performance for specific applications, it is a growing tendency to improve classifying performance by a combination of classification methods [12].

How to solve the multi-classification problems is challenging. There are two ways to deal with it [13]. Nilsson et al. [14] first use combination of binary classifiers to solve the multi-classification problems. The other way is that directly extends binary classifier into multiple classifier.

The main contributions of this paper are as following:

1. We compare twelve most commonly used algorithms for multi-classification in several aspects such as principle, important parameters, running time performance, and etc.

2. This paper considers the impact of differences of type and size in data sets. We chose binary classification and multi-classification, a small amount of data and a large size set as the evaluation data sets.

3. This paper gives in-depth analysis of multi-category classification for each class.

4. In this paper, we investigate the relationship between the single classifier and a combination of single classifier.

This paper discusses the 12 multiple classifiers: decision trees, random forests, extremely randomized trees, multi-class adaboost, stochastic gradient boosting, support vector machines (including linear and nonlinear), K nearest neighbors, multi-class logistic classifier, multilayer perceptron, naive Bayesian classifier and conditional random fields classifier.

The paper is organized as follows: section 1 is an introduction of studying content. In section 2 we discuss the related works. Overview of algorithms are discussed in section 3. Section 4 presents an experimental setup and parameters settings. In Section 5, we explain experimental results. Section 6 concludes the paper.

2 Related works

There are few comprehensive comparisons of multi-classification algorithms. King et al. [15] is the most comprehensive and earliest study of multi-classification algorithms including CART, the traditional algorithm C4.5, Naïve Bayes, K nearest neighbor, neural networks, and etc. However after that a few emerging algorithms such as support vector machines, random forests have been widely used. Further, data sets they used are too small while comparing to current big data. Then again their evaluation criteria is simple. Bauer et al. [16] thoroughly compare voting classification algorithms including bagging, boosting and its improved versions, but it is only comparing these two types of voting algorithm. LeCun et al. [17] use accuracy, rejecting rates and running time as the evaluation criteria. They compare algorithms: K-nearest neighbor, linear classifier, the main ingredient and polynomial classifiers on handwritten recognition data set. But only one data set used, it is not sufficient to evaluate different application scenarios. Ding et al. [18] use neural networks and support vector machines for protein test data set. They do detailed comparison of the accuracy of different parameters, but this comparison is relatively simple and data set is small. Tao et al. [19] study the decision trees, support vector machines, K nearest neighbor classification of gene sequences of organization application. However, this comparison only discusses single dataset. Foody et al [20] study multi-class image classification by support vector machines. But they only compare support vector machines, even linear support vector machine is not involved. Chih-Wei Hsu et al. [21] also study multi-class support vector machines for a more in-depth theoretical analysis and comparison, but is limited to multi-class support vector machine. Caruana et al. [22] study supervised learning algorithm (support vector machines, neural networks, decision trees, and etc.) in 9 different criteria such as ROC area, F evaluation and etc., but the literature is only discussed two classification data sets. Krusienski et al. [23] compare the Pearson correlation method, Fisher linear discriminant, stepwise linear discriminant, linear support vector machines, and Gaussian kernel support vector machines on P300 Speller data set. However, the data set is relatively simple, and small size of data is often difficult to compare running performance of support vector machine between linear and non-linear scenarios.

Table 1 lists the current situation of classifier comparisons.

Reference	Comparison of algorithms	Data sets	Evaluation criteria	Research domains
King et al.[15]	Symbolic learning(CART, C4.5, New ID, AC^2, Cal5, CN2) , statistic learning(Bayesian network, K-nearest, kernel density, linear discrimination, quadratic discrimination , logistic regression) , neural network	Satellite image, hand written digits, vehicle, segment, Credit risk, Belgian data, Shuttle control data, Diabetes data, Heart disease and head injury, German credit data	Accuracy	General purpose
Bauer et al[16	Bagging, boosting and its variants	Segment, DNA, chess, waveform, sat-image, mushroom, nursery, letter, shuttle	Average error rate, variance, bias	General purpose
Ding et al.[18	Support decision vector, neural network	Protein test dataset	Accuracy, Q-percentage, Accuracy	Bioinformatics
Tao et al.[19]	Support decision vector, Bayesian network, K-nearest, decision tree	ALL, GCM, SRBCT, MLL-leukemia, lymphoma, NCI60, HBC	Accuracy	Bioinformatics
Foody et al.[20]	Support decision vector, decision tree, discriminating analysis, neural network	Airborne sensor data	Accuracy,	Remote imaging
Chih-Wei Hsu et al[21]	Support decision vector	Iris, wine, glass, vehicle, segment, DNA, sat-image, letter, shuttle	Accuracy, running-time	General purpose

Caruana et al.[22]	Support decision vector, neural network, decision tree, memory based learning, bagged tree, boosted tree, boosted stumps	RMS, MXE, CAL, ADULT.	Accuracy, square error, inter-class cross entropy, ROC regions, F evaluation, recall and precision, average precision and recall, lift, probability calibration	General purpose
Krusienski et al.[23]	Pearson related methods, Fisher linear discrimination, stepwise linear discrimination, linear support decision vector, Gaussian kernel support decision vector	P300 Speller	ROC curve	Medicine domain
Our method	Decision tree, random forests, extremely randomized trees, multi-class adaboost classifier, multi-class logistic classifier, stochastic gradient boosting, multilayer perceptron, K nearest neighbors, naive Bayesian classifier and support vector machines(including linear and nonlinear)	SPECTF, Ionosphere, spam, optdigits and Scene 15	Overall accuracy, average precision and recall, average Jaccard, inter-class F and Jaccard evaluation, running performance.	General purpose

Table 1: Comparisons of multi-class classification algorithms.

Table 1 shows that the majority of current surveys focus on multi-class classifiers in a particular field, such as medicine, biology, remote sensing images, and etc. The data sets and methods used for evaluation is relatively simple. King et al's evaluation [15] is more comprehensive, however the comprising algorithms are classical. After that new algorithms emerge. Our comparing algorithms are the newest and most representative of the current tendency in a variety of application domains.

3 Overview of Algorithms

3.1 Brief introduction of algorithms

In order to better understand various classifiers to compare, we briefly introduce multiple classifier algorithms.

3.1.1 Decision tree

In machine learning, decision trees [24] is a predictive model. A decision tree is a flowchart-like structure in which internal node represents test on an attribute, each branch represents outcome of test and each leaf node represents class label, and then decision is taken after computing all attributes. A path from root to leaf represents classification rules. Decision tree is actually an adaptive basis function model [25], and can be expressed as follows

$$f(x) = E[y \mid x] = \sum_{m=1}^{M} w_m I(x \in R_m) = \sum_{m=1}^{M} w_m \text{¤}(x; v_m) \quad (1)$$

Where R_m is the $m'th$ region, w_m is the mean response in this region, and v_m encodes the choice of variable to split on, and the threshold value, on the path from the root to the $m'th$ leaf.

Classification tree is an ancient method, it has various variants, typically such as[26] ID3 (Iterative Dichotomiser 3) proposed by Ross Quinlan, is a greedy approach that in each iteration choose the best attribute value to split the data, but this method has the problem of local optimum. C4.5 also proposed by Ross Quin lan, is an improved ID3, can be used for classification. CART(classification and regression Trees)[27] proposed by Breiman has same process with C4.5 algorithm, except that the C4.5 uses an information entropy rather than Geni coefficient used by CART.

3.1.2 Random forests

Random forests are proposed by Leo Breiman and Adele Cutler[28]. It is an ensemble learning method for classification (and regression) that is built by constructing a multitude of decision trees at training time and outputting the class by voting of individual trees.

Random forests are a method of building a forest of uncorrelated trees using a CART like procedure, combined with randomized node optimization and bagging [29].

Random forests have the advantages of computing efficiency, improving the prediction accuracy without significantly increase of computational cost. Random forest can be well predicted up to thousands of explanatory variables [30], known as one of the best current algorithms [31].

3.1.3 Extremely randomized trees

Extremely randomized trees have been introduced by Pierre Geurts, Damien Ernst and Louis Wehenkel[32]. The algorithm of growing extremely randomized trees is similar to random forest, but there are two differences:

1. Extremely randomized trees don't apply the bagging procedure to construct a set of the training samples for each tree. The same input training set is used to train all trees.

2. Extremely randomized trees pick a node split very extremely (both a variable index and variable splitting value are chosen randomly), whereas random forests find the best split (optimal one by variable index and variable splitting value) among random subset of variables.

3.1.4 Multi-class adaboost classifier

Boosting has been a very successful technique for solving the two-class classification problem [33]. In going from two class to multi-class classification, most boosting algorithms have been restricted to reducing the multi-class classification problem to multiple two-class problems [34].

3.1.5 Stochastic gradient boosting

Gradient boosting proposed by Friedman [35] is a method to improve basic boosting algorithm. The traditional boosting method is adjusted weights to correct classification samples and error samples based on gradient descend at each iteration. The major difference of gradient boosting from the traditional boosting method is that purpose at each iteration is not to reduce the losses, but in order to eliminate the loss. The new model at each iteration is based on the residuals of former process. Inspired by Breiman[6]'s randomized bagging idea, Friedman introduces stochastic gradient boosting by randomized down-sampling to train basic classifier.

3.1.6 Support vector machines

Support vector machines [36, 37] is the method that mapped feature vector into a higher dimensional vector space, where a maximum margin hyper-plane is established in this space. So we choose the hyper-plane so that the distance from it to the nearest data point on each side is maximized. The greater the distance between the nearest data of different classes is, the smaller the total error classification is. The multi-class support vector machines [21] can be defined as follows

Supposing l groups sample: $(x_1,c_1),\dots,(x_l,c_l)$, where $x_i \in R^n, i=1,\dots,l$, $c_i \in \{1,\dots,k\}$ is the type of x_i . The $i-th$ support vector machine solve this problem.

$$\min_{w^i,b^i,\P^i} \quad \frac{1}{2}(w^i)^T w^i + C\sum_{j=1}^{t}\P_j^i (w^i)^T \tag{2}$$

$$(w^i)^T \text{¤}(x_j) + b^i \ge 1 - \P_j^i, if \;\; c_j = i \tag{3}$$

$$\P_j^i \ge 0, \quad j = 1,\dots,l \tag{4}$$

Where training sample x_i is mapping into high dimensional space by function ¤ and regularization parameters C . Minimizing $(1/2)(w^i)^T w^i$ means that we have to minimize $2/\|w^i\|$, the margins between two group data. The penalty $C\sum_{j=1}^{t}\P_j^i (w^i)^T$ is to reduce the number of training error. The core concept of support vector machine is to seek a balance between the regularization term $(1/2)(w^i)^T w^i$ and the training errors.

We get k decision functions after solving formula (4) .

$$\begin{gathered}(w^1)^T \text{¤}(x) + b^1 \\ \vdots \\ (w^k)^T \text{¤}(x) + b^k\end{gathered} \tag{5}$$

After that, we have the largest value of decision functions as the predictive class of x .

3.1.7 Linear support vector machines

SVM uses a nonlinear mapping that converts low-dimensional feature space into a high-dimensional space to get better discriminative. However, in other applications, the input feature itself is a high-dimensional space, if more is mapped to more high latitudes, it may not be able to get better performance. Their own space can be directly used as identification. The linear support vector machine SVM [38] is suitable for this scenario. For multi-classification, Crammer et al. [39] propose this method to solve the problem. We define the original question as follow

Supposing the training data set $T=\{(x_1,c_1),(x_2,c_2),\dots,(x_N,c_N)\}$, where, $x_i \in \text{¡} \subseteq R^n$ is feature vector, $y_i \in Y = \{1,2,\dots,k\}$ is the type of instance, $i=1,2,\dots,l$. The multi-class problem can be formulated as the following primal problem.

$$\begin{aligned}&\min_{\{w_m\},\{\P_i\}} \frac{1}{2}\sum\nolimits_m \|w_m\|^2 + C\sum\nolimits_i \P_i \\ &s.t. \quad w_{y_i}^T x_i \ge e_i^m - \P_i \quad \forall_m, i,\end{aligned} \tag{6}$$

where, $C>0$ is the regularization parameter, w_m is the weight vector associated with class m , $e_i^m = 1 - \pm_{c_i,m}$, and $\pm_{c_i,m} = 1$ if $c_i = m$, $\pm_{c_i,m} = 0$ if $c_i \ne m$. Note that in (7), the constant $m = c_i$ corresponds to the non-negativity constraint: $\P_i \ge 0$. The decision function is

$$\arg\max_m w_m^T x \tag{7}$$

3.1.8 K nearest neighbors

K nearest neighbours algorithm (or k-NN for short) is a non-parametric method used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space.

In k-NN classification, the output is a class membership. An object is classified by a majority vote of its neighbours, with the object being assigned to the class most common among its k nearest neighbours (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbour.

3.1.9 Multi-class logistic classifier (maximum entropy classifier)

In some cases, multi-class logistic regression well fits features. It has the formula [25]

$$p(y=c \mid x,W) = \frac{\exp(w_c^T x)}{\sum_{c'=1}^{C}\exp(w_{c'}^T x)} \tag{8}$$

Where, $p(y=c \mid x,W)$ is the predictive probability. y is the class type of totally C . w_c is the weight of class c, and approximated by maximum posterior probability. With this, the log-likelihood has the form

$$\begin{aligned} l(W) &= \log\prod_{i=1}^{N}\prod_{c=1}^{C}\text{«}_{ic}^{y_{ic}} = \sum_{i=1}^{N}\sum_{c=1}^{C} y_{ic}\log\text{«}_{ic} \\ &= \sum_{i=1}^{N}\left[\left(\sum_{c=1}^{C} y_{ic} w_c^T x_i\right) - \log\left(\sum_{c'=1}^{C}\exp(w_{c'}^T x_i)\right)\right]\end{aligned} \tag{9}$$

Where, $\text{«}_{ic} = p(y_i = c \mid x_i, W)$. This model can be optimized by L-BFGS[41].

3.1.10 Multilayer perceptron

A multilayer perceptron (MLP) is a feed forward artificial neural network model that maps sets of input data onto a set of appropriate outputs. A MLP consists of multiple layers of nodes in a directed graph, with each layer fully connected to the next one. Except for the input nodes, each node is a neuron (or processing element) with a nonlinear activation function.

MLP[57] has evolved over the years as a very powerful technique for solving a wide variety of problems. Much progress has been made in improving performance and in understanding how these neural networks operate. However, the need for additional improvements in training these networks still exists since the training process is very chaotic in nature.

3.1.11 Naive Bayesian classifier

A naive Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions. A more descriptive term for the underlying probability model would be “independent feature model”

Assuming sample X , belongs to type C_i . The class-conditional density is

$$P(C_i \mid X) = \frac{f(X \mid C_i)P(C_i)}{p(X)} = \frac{f(X \mid C_i)P(C_i)}{\sum_{j=1}^{n} f(X \mid C_j)P(C_j)} \qquad (10)$$

Where, $f(. \mid C_j)$ is the maximum likelihood. Input features is x , c is class type.

A simpler alternative to generative and discriminative learning is to dispense with probabilities altogether, and to learn a function, called a discriminant function, that directly maps inputs to outputs. The decision function of naive Bayesian classifier is

$$c = \arg\max_{c_k} P(\mathrm{C} = c_k)\prod_{j=1}^{n} P(\mathrm{Y}_i = \mathrm{y}_{\mathrm{j}} \mid \mathrm{C} = c_k) \qquad (11)$$

3.1.12 Conditional random fields classifier

CRFs (Conditional random fields) are a class of statistical modelling method often applied in pattern recognition and machine learning, where they are used for structured prediction. Whereas an ordinary classifier predicts a label for a single sample without regard to "neighboring" samples, a CRF can take context into account; e.g., the linear chain CRF popular in natural language processing predicts sequences of labels for sequences of input samples.

We define a CRF on observations X and random variables Y as follows:

Let $G = (V , E)$ be a graph such that $Y = (\mathrm{Y}_{\mathrm{v}})_{v \in V}$, so that Y is indexed by the vertices of G . Then (X, Y) is a conditional random field when the random variables Yu, conditioned on X, obey the Markov property with respect to the graph:

$$p(\mathrm{Y}_{\mathrm{v}} \mid \mathrm{X}, \mathrm{Y}_{\mathrm{w}}, \mathrm{w} \neq \mathrm{v}) = \mathrm{p}(\mathrm{Y}_{\mathrm{v}} \mid \mathrm{X}, \mathrm{Y}_{\mathrm{w}}, \mathrm{w} - \mathrm{v}) \qquad (12)$$

Where $w \square v$ means that w and v are neighbors in G. What this means is that a CRF is an undirected graphical model whose nodes can be divided into exactly two disjoint sets X and Y , the observed and output variables, respectively; the conditional distribution $p(\mathrm{y} \mid \mathrm{x})$ is then modeled. For classification problem, we compute the maximum conditional probabilistic distribution.

3.2 Comparison of algorithms

With reference to [45] for the multi-classification, we made a comparison of 12 algorithms as shown in table 2.

We analyze and summarize in Table 2 as follow:

1. Algorithms type

Aside from Naive Bayes, others are discriminant model. Bayesian algorithm by learning the joint distribution $P(C, X)$, then obtains the conditional probability $P(X \mid C)$. The classifying prediction is achieved by maximizing likelihood approximation. However the discriminant method directly makes prediction by the discriminant function or conditional probability.

2. Algorithms trait

Decision trees, random forest, extreme random tree, multi-class adaboost upgrade and stochastic gradient boosting all belong to model with adaptive basis functions that can be grouped into common additive model[25], as shown in Equation 13.

$$f(x) = \sim + f_1(x_1) + \ldots + f_D(x_D) \qquad (13)$$

Where f_i is the sub-model obtained through training sample. $f(x)$ is the superposition of sub-models. Decision tree is the basic sub-model for tree-like algorithms. Upon whether the use of all samples for training, these kinds of algorithms can be divided into random forests and extremely random tree. Random forests is to build sub-model through random bagging sampling. However, extremely random tree obtains sub-model using all training samples, but randomly selecting the splitting threshold. Multi-class adaboost and stochastic gradient boosting is a linear combination of sub-models (weak classifiers). The difference lies in their learning algorithms.

Multilayer Perceptron, linear and non-linear support vector machines can be classified as kernel methods [46]. The unified formula has the form

$$f(x) = w \cdot \Phi(x) + b \qquad (14)$$

where w is real weight , $b \in R$ is bias. $\Phi(x)$ function is the type of the classifiers, for MPL $\Phi(.) = (\Phi_1(\cdot), \Phi_1(\cdot), \ldots, \Phi_N(\cdot))$, the $i-th$ hidden is defined as $\Phi_i(x) = h(v_i x + d_i)$. h is the mapping function , generally a hyperbolic function or shape function B is chosen by MPL. For linear support vector machine $\Phi(x)$ is linear function, rather than polynomial function, Gaussian kernel function, and etc. for non-linear support vector machine.

K nearest neighbor model is constructed according to the division of distance relationship of the training feature space, being classified by a majority voting.

Multi-class logistic regression (maximum entropy) is the probabilistic choice model with constraints that uncertain contents are treated with equal probability of using entropy maximization to represent.

Naive Bayes classifier is based on the conditional independence assumption of training samples, learning parameters with the joint probability distribution though Bayes' theorem, then classifying a given sample, by maximizing the posterior probability to get corresponding class.

3. Learning policy, loss and algorithms

Decision trees, random forests, extremely randomized trees belong to the maximum likelihood approximation of learning strategies, with the loss of log-likelihood function.

Algorithms	Algorithms type	Algorithms characteristic	Learning policy	Loss of learning	Learning algorithms	Classification strategy
Decision tree	Discriminant	Classification tree	Regularized maximum likelihood estimation	Logarithmic likelihood loss	Feature selection, generation, prune	IF-THEN policy based on tree spitting
Random tree	Discriminant	Classification tree(based on bagging)	Regularized maximum likelihood estimation	Logarithmic likelihood loss	Building multi-decision tree based on bagging of subsampling	Sub-tree voting
Extremely random tree	Discriminant	Classification tree	Regularized maximum likelihood estimation	Logarithmic likelihood loss	Building multi-decision tree	Sub-tree voting
Multi-class adaboost	Discriminant	Linear combination of weak classifier(based on decision tree)	Addition minimization loss	Exponent loss	Forward additive algorithm	Linear combination of weighted maximum weak classifiers
Stochastic gradient boosting	Discriminant	Linear combination of weak classifier(based on decision tree)	Addition minimization loss	Exponent loss	Stochastic gradient descent algorithm	Linear combination of weighted maximum weak classifiers
Non-linear Support vector machine (based on libsvm)	Discriminant	Super-plane separation, kernel trick	Minimizing the loss of regular hinge, soft margin maximization	Hinge loss	Sequential minimal optimization algorithm (SMO)	Maximum class of test samples
Linear SVM (based on liblinear)	Discriminant	Super-plane separation	Minimizing the loss of regular hinge, soft margin maximization	Hinge loss	Sequential dual method	Maximum weighted test sample
K-nearest	Discriminant	Distance of feature space				Multiple voting, empirical loss minimization
Multi-logistic (Maximum entropy)	Discriminant	Conditional probabilistic distribution, Log-linear model	Regularized maximum likelihood estimation	Logistic loss	L-BFGS	Maximum likelihood estimation
Multilayer perceptron	Discriminant	Super-plane separation	Minimization of error separation distance point to the hyper-plane	Error separation distance point to the hyper-plane	Random gradient decrease	Maximum weighted test sample
Naive Bayesian classifier	Generative	Joint distribution of feature and class, conditional independent assumption	Maximum likelihood estimation, Maximum posterior probability	Logarithmic likelihood loss	Probabilistic computation	Maximum posterior probability
Conditional Random Fields	Discriminant	Conditional probabilistic distribution under observing sequence, Log-linear model	Maximum likelihood estimation, Regularized maximum likelihood estimation	Logarithmic likelihood loss	Random gradient decrease, quasi-newton methods	Maximum likelihood estimation

Table 2: Summary of twelve multi-class methods.

The decision tree's optimal strategy is to learn some features through a recursive process and split the tree in accordance with the feature of the training samples. In order to have a better generalization ability, the decision tree model has to be pruning for removal of over-fitting.

Random forests is based on random sampling based on the approach (bagging) to form more stars forest trees. Extremely randomized trees is randomly selecting the splitting value to build decision trees forests.

Multilayer Perceptron, linear and non-linear support vector machines are to separate hyper-plane. The difference is that multilayer perceptron is to minimize the error hyper-plane, however linear and non-linear support vector machine is a minimal loss of hinge page. Perceptron learning algorithm is stochastic gradient descent, linear support vector machine is a sequential dual method, and non-linear support vector machine is a sequential minimal optimization method.

K-nearest neighbor is based on distance of feature space.

Multi-class logistic classifier (maximum entropy) learning strategies can be seen as either maximum likelihood estimation, or a logical loss minimization. Loss of function is a linear logarithmic function. The model learning is the maximum likelihood estimation or regularized maximum likelihood estimation under certain training conditions. Optimization algorithm is L-BGFS.

Naive Bayesian probability is a typical generative model. The maximum likelihood estimation and Bayesian posterior probabilities is calculated based on the joint probability and conditional independence assumption.

4. Classification strategy

Decision tree uses the IF-THEN rules for classification decisions based on the value of the model learned. Random forests and extremely random tree is based on the voting of every single decision tree classification, then taking the highest vote as the final results.

Multilayer Perceptron, multi-class logistic regression (maximum entropy), linear and non-linear support vector machine have the same form of classification decisions

$$class\ \ of\quad x \equiv \arg\max_{i=1,\ldots,k}((w^i)^T \text{¤}(x) + b^i). \tag{15}$$

The difference lies in the choice of $\text{¤}(x)$. Multilayer perceptron machine chooses B-shaped function, hyperbolic tangent function, and etc.; Log-linear functions is chosen for multi-class logistic regression; linear support vector machine choses a linear function; nonlinear support vector machine's choice is non-linear kernel function.

K nearest neighbor is a majority voting that output classification is determined by choosing K nearest voting in light of test sample' s distance to the learned model.

Naive Bayesian decision strategy is the rule of maximizing the posterior probability.

4 Experimental Setup

In order to evaluate the performance of various types of classifiers, we implemented our comparisons based on Darwin [47], Opencv[56], Libsvm [48] and liblinear [11] in C++. This paper compared 12 kinds of algorithms.

4.1 Performance comparisons

Confusion matrix (Confusion Matrices) [49, 50] is a common performance evaluation method in pattern recognition classification, which characterizes the relationship between the type of real classes and the recognition classes. For multi-classification problem (For simplifying the representation, we take three categories as example) is illustrated in Table 2.

Prediction class True class	A	B	C
A	AA	AB	AC
B	BA	BB	BC
C	CA	CB	CC

Table 3: Statistics of confusion matrices for samples classification.

Where A, B and C are three classes, AA, BB and CC represent the correct prediction number of samples, the remaining number of samples is representative of the error prediction. AA represents the number of samples correctly identified as samples A. AB is predictive number that original Sample A which is incorrectly predicted as Sample B. The remaining items have the same meaning.

Total accuracy rate can be calculated based on confusion matrix as follows.

$$TA = (AA + BB + CC) / (AA + AB + ... + CC) \tag{16}$$

Where $AA + AB + ... + CC$ is the total number of sample. TA is the total accuracy.

Precision and recall are quantitative evaluation method. They are not only used to evaluate the accuracy of each class, but also the important standard to measure the performance of the classification system.

Precision is the fraction of retrieved instances that are relevant. Precision reflects the classification accuracy. In practical applications, the average precision are often used to evaluate multi-classification (taking categories as example), which is calculated as follows

$$avgprecision = (AA / (AA + BA + CA) + BB / (BA + BB + BC) + CC / (AC + BC + CC)) \tag{17}$$

Recall is the fraction of relevant instances that are retrieved. Recall reflects the classification comprehensiveness. In practical applications, the average recall are often used to evaluate multi-classification (taking categories as example), which is calculated as follows

$$avgrecall = (AA / (AA + AB + AC) + BB / (BA + BB + BC) + CC / (CA + CB + CC)) \tag{18}$$

F_1 Measure is an integrated measurement method of the recall and precision. Higher values reflect the recall and precision better integrated. F_1 is defined as

$$F_1 = 2 * \frac{avgprecision * avgrecall}{avgprecision + avgrecall} \tag{19}$$

Jaccard Coefficient [51] is used for comparing the similarity and diversity of sample sets. Taking class A as example, the formula is

$$JC_A = \frac{AA}{AA + AB + AC + BA + CA} \tag{20}$$

Average Jaccard coefficient reflects the average of the various categories Jaccard coefficient, which is calculated as

$$avgJC_A = (\frac{AA}{AA + AB + AC + BA + CA} + \frac{BB}{AB + BB + CB + BA + BC} + \frac{CC}{CA + CB + CC + AC + BC})/3 \tag{21}$$

Jaccard coefficient predicts more accurately reflects higher. Jaccard coefficient can evaluate multi-class classification in each class.

4.2 Data sets and data transformation

We evaluate the performance of these algorithms on five data sets that consists of three binary classification data sets and two multi-classification data sets.

4.2.1 SPECTE Heart data set

SPECTF[52] means single-photon emission computed tomography cardiac data sets. SPECTF is a new data set[53]. In cleaning process, the records with missing information, incomplete picture records are filtered, since the original image scale is not uniform, so the original image is transformed into gray image (0 to 255).

After cleaning, the data set contains 44 features that record 22 region of interest for the cardiac systolic and expanded state. The data type is an integer type (rang from 0 to 100). The data set has total 267 instances of which containing 80 training samples and 187 test samples. Each instance has two states: normal and abnormal. In experiment, we converted class label into class 0 (normal) and class 1 (abnormal). Histogram distribution of training samples showed normal shape is shown in Figure 1 (a).

4.2.2 Ionosphere data set

Ionosphere data set[52] includes is a set of radar data sets created by a military system acquisition in Labrador NATO airbase, Goose Bay, Canada.

The data set has total of 34 feature (integer, decimal type) that record 17 sub-pulses labels and values. The data set consists of 351 instances of which 100 instances are training samples, 251 instances are test samples. Each instance has two states: good and bad. In the experiment we converted class label into class 0 that represents the existence of the facts) and class 1 the non-existent facts. Feature type of the dataset is real type, range from -1 to

a. Histogram of the distribution of the training samples is shown in Figure 1 (b).

4.2.3 Spam data set

Spam[52] is a spam filtering data set that has total of 57 data features. 48 consecutive real number is used to represent the percentage of messages of 48 words (i.e. make, address, all, conference) and 6 word occurrences (!, (, [,, $, #) in email.

The data set has a total of 4601 instances, where the training instances are 3065, testing instances are 1,536. Each instance has two states: 0 for not spam and 1 for junk mail. The data set is real type feature information, the range is 0 to 9090. The training sample distribution histogram is scattered, as shown in Figure 1 (c).

4.2.4 Optidigits data set

Optdigits[52] is the information collection extracted from 43 individuals (30 of them for training, 13 as a testing) by the handwritten Optical Character Recognition bit image Standardization American National Standards Institute of Technology (NIST).

The data set has a total of 5620 instances, where the training instances are 3823, test instances are 1797. The instances have ten classes, from 0 to 9 digits. Feature data type is a positive integer, ranging from 0 to 16. Histogram distribution of training samples is shown in Figure 1 (d).

4.2.5 Scene 15 data set

Scene 15[54] data set is a post-processing data set. The original data set has total of 15 classes, 4485 images. We randomly selected sample of 2250 as training samples, 2235 samples for testing.

We used the method of PHOW (Pyramid Histogram Of visual Word) [54] for feature extraction. 12000 descriptors were obtained for each image. In order to obtain a better classification within the features, we used method proposed by vedaldi[55] for features kernel transformation, finally we got 36,000 dimensional feature descriptors to characterize a single image. In the experiment we converted 15 classes into class 0 to 14, representing the bedroom, CALsuburb, industrial, kitchen, livingroom, MITcoast, MITforest , MIThighway, MITinsidecity, MITmountain, MITopencountry, MITstreet, MITtallbuilding, PARoffice and store respectively. Feature data type is a positive real number, the range is -0.3485 to 0.4851, further feature data is relatively concentrated, similar to normal training. Histogram distribution of training samples is shown in Figure 1 (e).

This data set has large dimensions of features and a large size of data. Selection of this data set is mainly to test algorithms adaptability for real application scenarios.

In Figure 1, we demonstrated our diversity of data sets. SPECIF, ionoispher and spam are small scale data sets. They are for binary classification. Optigits and Scene 15 are large scale data sets. They are for multiclass classification. At the same time, the range of our feature vector value are large.

The range of Figure 1 feature values is also huge. The range of SPECIF is from 0 to 100, ionoispher from -1 to 1, and spam from 0 to 1000, Optigits from 0 to 20 and Scene 15 from -0.5 to 0.5. The diversity for our comparing experiments are necessary. This shows that our experiments are rich and valuable.

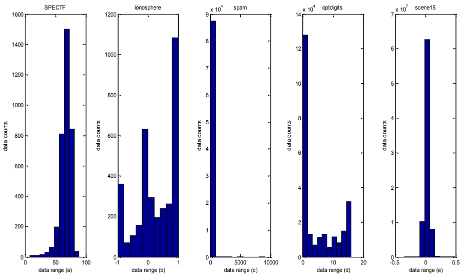

Figure 1: The histogram characteristics of training data sets.

4.2.6 Dataset selection

We select the data set is based on the following considerations.

1. Both binary classification and multi-classification data sets were taken into account for the purpose that some algorithm are superior to binary classification, while others are more suitable for multi-classification problems.

2. For better evaluation of the adaptability of the algorithms, we chosen both low dimensional data set with small size and high dimensional data set with large size.

3. We also paid attention to the data type differences, both pure integer and double data type were included. There were also real data type features, which had either wholly positive or negative features.

4. It can be seen in Figure 1 that the histogram of the distribution of the data we have chosen the difference is quite different. Some are concentrated, just like normal distribution, while others are relatively sparse.

4.3 Parameter setting and selection

4.3.1 Decision tree

1. The maximum depth of tree: The default value was set to 1. With the value increase, classification accuracy and running time will increase. We set maximum depth to 2, 4, 6, 8, 10, 12, 14, 16, 18 and 20 respectively. We found the highest accuracy rate when the maximum depth is 10.

2. Splitting Criteria Consideration: We tested the wrong classification rule, entropy rule, and Geni coefficient rule respectively. We found that Geni coefficient got the best of all, and mistake classification rule got the worst of all.

3. The first split minimum number of samples were tested with value of 0, 2, 4, 6, 8 and 10 respectively. We found the minimum number of samples was increased, but performance degraded.

4. The maximum number of features of the training sample, the default was set to 1000.

4.3.2 Random forests

1. A maximum depth of random forests, as the same of decision tree, the optimal value was set to 10.
2. The maximum number of training samples feature, the default was set to 1000, the value should change with the number of sample features change, as long as the sample itself to adapt to the largest number of features.
3. The number of decision trees: we tested the value of 20, 40, 60, 80, 100, 120, 140, 160, 180 and 200 respectively. We found that the number increases, the execution time also increases, and after over the value of 100, the improvement of the accuracy rate is not obvious. So we set it to 100.
4. The accuracy of the random forest was used to control the iteration. We tested the value of 0.0001, 0.001, 0.01 and 0.1 respectively. We found that the smaller the value was, the longer the execution time was, and after over the value of 0.001, the accuracy had no substantially change. We set it to 0.001.

4.3.3 Extremely randomized trees

Parameters were consistent with the random forest.

4.3.4 Multi-class adaboost classifier

1. For boosting methods, we tested discrete boosting, gentle boosting and real boosting respectively, found that gentle boosting is the best of all.
2. Tree depth was initialized with value of 10.
3. Shrinkage factor: we set it with value of 0.1, 0.2, 0.1 and 0.4 respectively, and found that 0.1 was the best of all.
4. Maximum boosting number: we set it to the value of 10, 50, 100, 200 and 400 respectively. We found that after over the value of 100, the precision had no significant increase. So we set it to value of 100.

4.3.5 Stochastic gradient boosting

1. Maximum depth of the tree, same as in decision tree, was set it to 10.
2. Loss function type for classification problems was generally selected for cross-entropy loss.
3. Shrinkage factor was set to 0.1.
4. Subsampling percentage was used to control the sampling percentage for every single tree in the training process, we set it to 0.8.
5. Maximum lift was set to100.

4.3.6 Support vector machines

1. Type: we set it to C - Support Vector Classification and v - support vector classification, it was found that v - support vector classification is better than C - support.

Penalty factor for C - support vector classification was set to 1.

Penalty factor for v -support vector classification, value range was $[0,1]$, we set it to 0.5, p was set to 0.1.

2, Kernel types are linear, radial basis, sigmoid-type function, POLY.

Data sets Total Accurac Method		SPECTF	Ionosphere	Spam	optdigits
C - SVM	POLY	**0.770**	**0.565**	**0.618**	**0.978**
	RFB	0.898	0.749	0.772	0.562
	Sigmoid function	0.080	0.693	0.374	0.101
v - SVM	POLY	0.748	0.733	0.629	0.934
	RFB	0.903	0.796	0.779	0.586
	Sigmoid function	0.080	0.840	0.648	0.101

Table 4: Overall accuracy of libSVM on four data sets.

From table 4, we tested non-linear support vector machines total accuracy on four data sets. We selected v - Support Vector POLY for classification as a non-linear support vector machines kernel type.

4.3.7 Linear support vector machines

From table 5, we can see that L2R_L2LOSS_SVC for loss function got good accuracy, so we chose L2R_L2LOSS_SVC loss function as loss type of linear support vector machines.

Total accuracy Data sets Cost type	SPECTF	Ionosphere	Spam	Optdigits
L2R L2LOSS SVC	**0.620**	**0.856**	**0.898**	**0.947**
L2R L2LOSSSVC DUAL	0.577	0.856	0.617	0.939
L2R L1LOSS SVC DUAL	0.577	0.848	0.865	0.935
MCSVM CS	0.805	0.860	0.631	0.933

Table 5: Overall accuracy of liblinear on four data sets.

4.3.8 K nearest neighbors

Nearest neighbor K was set to the value of 1, 3, 5, 7, 9, 11, 13 and 15 respectively. The accuracy was found to reduce with increasing K value, so we set it to 1.

4.3.9 Multi-class logistic classifier

According to our single test with variable regularization factor values of 1.0e-12, 1.0e-11, 1.0e-10, 1.0e-9, 1.0e-8, 1.0e-7 and 1.0, we found that the best results is the regularization factor with value of 1.0e-9. So in our comparing experiments, the regularization factor (REG_STRENGTH) was set to 1.0e-9.

Similarly, we did same experiment, most optimum results were found with the iteration range from 800 to 980. So in our comparing experiments, maximum number of iterations was set to 1000 for avoidance of losing optimum values.

4.3.10 Multilayer perceptron

1. The propagation algorithm was backward and forward propagation respectively. Apparently backward propagation algorithm achieved significantly higher accuracy.
2. Gradient weight was typically set to 0.1.
3. The momentum, front weights reflecting differences in two iterations, was typically set to 0.1.

4.3.11 Naive Bayesian Classifier

No parameters needed to be set.

4.3.12 Conditional Random Field Classifier

We only need set the classification type in this experiments.

5 Experimental results and analysis

Firstly, we need define descriptions for the symbol in tables: DecisionTree represents decision tree classifier. RandomForest represents the random forests. ExtraTrees is extremely random tree. BoostedClassifier represents the multi-class adaboost classifier. GradientBoostTree represents stochastic gradient boosting, libSVM is support vector machines. libLinear represents linear support vector machine. Knearest represents K nearest neighbor classifier, MultiClassLogistic is multi-class logistic classifier. MultiLayerPerceptron represents multilayer perceptron.NormalBayesianNet represents the naive Bayesian classifier. CRF represents Conditional Random Fields classifier

5.1 Overall accuracy on five data sets

From table 6, on SPECTF data set, stochastic gradient boosting achieved the highest overall accuracy, average precision, recall and Jaccard value, following by non-linear support vector machine, random forest, adaboost classifiers and CRF have achieved relatively good performance.

Algorithm	Overall accuracy	Average precision	Average recall	Average jaccard coefficient
DecisionTree	0.657	0.5421	0.631	0.381
RandomForest	0.759	0.5721	0.686	0.456
ExtraTrees	0.668	0.5537	0.667	0.394
BoostedClassifier	0.716	0.5847	0.754	0.440
GradientBoostTree	**1.000**	**1.000**	**1.000**	**1.000**
libSVM	0.770	0.5664	0.662	0.458
libLinear	0.620	0.5342	0.611	0.356
Knearest	0.604	0.5666	0.724	0.362
Multi-classLogistic	0.620	0.5342	0.611	0.356
MultiLayerPerceptron	0.679	0.5377	0.612	0.391
NaiveBayesianNet	0.588	0.510	0.532	0.327
CRF	0.683	0.543	0.652	0.439

Table 6: Overall accuracy on SPECTF data set.

Algorithm	Overall accuracy	Average precision	Average recall	Average jaccard coefficient
DecisionTree	0.792	0.765	0.799	0.635
RandomForest	0.828	0.807	0.854	0.691
ExtraTrees	0.888	0.862	0.897	0.780
BoostedClassifier	0.900	0.876	0.899	0.798
GradientBoostTree	**1.000**	**1.000**	**1.000**	**1.000**
libSVM	0.749	0.735	0.775	0.584
libLinear	0.856	0.867	0.787	0.694
Knearest	0.844	0.822	0.804	0.691
Multi-classLogistic	0.768	0.732	0.750	0.593
MultiLayerPerceptron	0.768	0.742	0.775	0.603
NaiveBayesianNet	0.733	0.683	0.633	0.500
CRF	0.863	0.870	0.793	0.612

Table 7: Overall accuracy on Ionosphere data set.

From table 7, on Ionoshere datasets, stochastic gradient boosting achieved the highest overall accuracy, average precision, recall and Jaccard value, following by adaboost classifier, extremely randomized trees, CRF and linear support vector machines.

Algorithm	Overall accuracy	Average precision	Average recall	Average jaccard coefficient
DecisionTree	0.905	0.905	0.896	0.819
RandomForest	0.930	0.936	0.918	0.862
ExtraTrees	0.863	0.897	0.832	0.738
BoostedClassifier	0.940	0.942	0.933	0.882
GradientBoostTree	**0.981**	**0.981**	**0.980**	**0.962**
libSVM	0.618	0.687	0.522	0.332
libLinear	0.898	0.892	0.901	0.811
Knearest	0.775	0.765	0.764	0.622
Multi-classLogistic	0.923	0.921	0.918	0.852
MultiLayerPerceptron	0.908	0.904	0.905	0.827
NaiveBayesianNet	0.602	0.801	0.500	0.301
CRF	0.784	0.778	0.790	0.653

Table 8: Overall accuracy on spam data set.

From table 8, on the spam dataset stochastic gradient boosting accuracy achieved the highest overall average precision, recall and Jaccard value, following by multi-lass adaboost classifier, random forests. For binary classification problem but, under the low-dimensional data set, the stochastic gradient boosting and ensemble classifiers such as multi-class adaboost showed a high performance.

From table 9, on optdigits data set, for the multi-classification problem, stochastic gradient boosting achieved the highest overall accuracy, average precision, recall and Jaccard value. K-nearest neighbor, non-linear support vector machine, random forests, extremely randomized trees and CRF also achieved good overall performance. Linear support vector machines, many types of logic, multi-layer perceptron is manifested in ordinal performance, and decision trees and Naïve Bayes classifier achieved the worst performance of all.

Algorithm	Overall accuracy	Average precision	Average recall	Average jaccard coefficient
DecisionTree	0.847	0.850	0.847	0.738
RandomForest	0.966	0.966	0.966	0.936
ExtraTrees	0.969	0.970	0.969	0.942
BoostedClassifier	0.870	0.880	0.870	0.778
GradientBoostTree	**1.000**	**1.000**	**1.000**	**1.000**
libSVM	0.978	0.979	0.978	0.959
libLinear	0.947	0.948	0.946	0.901
Knearest	0.979	0.980	0.979	0.961
Multi-classLogistic	0.943	0.944	0.943	0.893
MultiLayerPerceptron	0.946	0.947	0.946	0.897
NaiveBayesianNet	0.843	0.881	0.844	0.732
CRF	0.960	0.962	0.958	0.932

Table 9: Overall accuracy on optdigits data set.

From table 10, due to the high dimensional feature data of Scene15 data set, multi-layer perceptron, naive Bayesian classifier and nonlinear support vector machine were failed. This mainly caused by the intrinsic shortcomes of nonlinear SVM, Bayesian networks and multi-layer perceptron. They have too many inner loops and intermediate phased which need computation and storage. This situation is worse when feature data is high dimensional. Another reason is that our testing environment is limited. If we have enough memory and strong CPU capability, I think this phenomenon will disappear. In another view, this also reveals that they

Algorithm	Overall accuracy	Average precision	Average recall	Average Jaccard coefficient
DecisionTree	0.397	0.391	0.376	0.246
RandomForest	0.531	0.548	0.493	0.339
ExtraTrees	0.655	0.639	0.637	0.479
BoostedClassifier	0.422	0.502	0.424	0.274
GradientBoostTree	**0.999**	**0.999**	**0.999**	**0.999**
libSVM	Invalid	Invalid	Invalid	Invalid
libLinear	0.815	0.813	0.8113	0.694
Knearest	0.565	0.602	0.5421	0.382
Multi-classLogistic	0.794	0.797	0.7915	0.665
MultiLayerPerceptron	Invalid	Invalid	Invalid	Invalid
NaiveBayesianNet	Invalid	Invalid	Invalid	Invalid
CRF	0.650	0.625	0.621	0.456

Table 10: Overall accuracy on Scene15 data set.

Algorithm	F_1		Jaccard coefficient	
	Class0	Class1	Class0	Class1
RandomForest	0.286	0.855	0.167	0.747
ExtraTrees	0.244	0.788	0.139	0.650
BoostedClassifier	0.312	0.822	0.185	0.697
GradientBoostTree	**1.000**	**1.000**	**1.000**	**1.000**
libSVM	0.271	0.863	0.157	0.760
libLinear	0.202	0.751	0.113	0.601
Knearest	0.260	0.730	0.149	0.575
Multi-classLogistic	0.202	0.751	0.113	0.601
MultiLayerPerceptron	0.211	0.799	0.118	0.665
NaiveBayesianNet	0.154	0.728	0.083	0.572
CRF	0.278	0.813	0.156	0.745

Table 11: Inter-class accuracy on SPECTF data set.

Algorithm	F_1		Jaccard coefficient	
	Class0	Class1	Class0	Class1
RandomForest	0.919	0.840	0.850	0.724
ExtraTrees	0.916	0.835	0.844	0.717
BoostedClassifier	0.926	0.847	0.863	0.734
GradientBoostTree	**1.000**	**1.000**	**1.000**	**1.000**
libSVM	0.796	0.674	0.661	0.508
libLinear	0.903	0.723	0.824	0.566
Knearest	0.890	0.735	0.802	0.581
Multi-classLogistic	0.827	0.651	0.706	0.482
MultiLayerPerceptron	0.820	0.678	0.695	0.513
NaiveBayesianNet	0.822	0.464	0.698	0.302
CRF	0.908	0.838	0.846	0.721

Table 12: Inter-class accuracy on Ionosphere data set.

Algorithm	F_1		Jaccard coefficient	
	Class0	Class1	Class0	Class1
RandomForest	0.944	0.908	0.894	0.831
ExtraTrees	0.897	0.798	0.814	0.664
BoostedClassifier	0.952	0.923	0.908	0.857
GradientBoostTree	**0.985**	**0.977**	**0.970**	**0.955**
libSVM	0.757	0.107	0.609	0.056
libLinear	0.914	0.878	0.841	0.783
Knearest	0.815	0.715	0.687	0.557
Multi-classLogistic	0.937	0.903	0.881	0.823
MultiLayerPerceptron	0.924	0.886	0.859	0.795
NaiveBayesianNet	0.752	0.002	0.603	0.001
CRF	0.933	0.891	0.885	0.838

Table 13: Inter-class accuracy statistics on Spam data set.

have tough condition for real application.

However, stochastic gradient boosting achieved the highest overall accuracy, average precision, recall and Jaccard value, following by linear support vector machines, multi-class logistic regression. Other algorithms got poor performance.

5.2 Inter class accuracy and jaccard coefficient evaluation on five data sets

The statistical results above reflected the overall performance of the algorithms. However, inter classes F_1 and Jaccard could reflect more detailed information. Accuracy of each class may vary greatly due to differences of the data. Overall each accuracy was determined by average of sum each class accuracy.

From table 11, on the data set SPECTF, class 0 represents normal, class 1 represents abnormal. Stochastic gradient boosting was fully recognized, so it had the highest F_1 and Jaccard coefficients in each sub class (both class 0 and class 1). Remains of algorithms' F_1 and Jaccard coefficients were not high in class 0, however there were high accuracy in class 1. This

Algorithm		Class0	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8	Class9
DecisionTree	F_1	0.933	0.830	0.813	0.817	0.803	0.891	0.940	0.836	0.777	0.832
	Jaccard	0.875	0.710	0.685	0.690	0.671	0.804	0.887	0.719	0.636	0.712
RandomForest	F_1	0.989	0.965	0.983	0.962	0.981	0.975	0.986	0.972	0.920	0.933
	Jaccard	0.978	0.933	0.966	0.926	0.962	0.952	0.973	0.945	0.851	0.875
ExtraTrees	F_1	0.992	0.963	0.994	0.956	0.986	0.975	0.989	0.972	0.938	0.934
	Jaccard	0.983	0.928	0.989	0.915	0.973	0.952	0.978	0.945	0.883	0.877
BoostedClassifier	F_1	0.960	0.804	0.930	0.813	0.899	0.919	0.960	0.850	0.783	0.798
	Jaccard	0.923	0.672	0.870	0.685	0.816	0.850	0.924	0.740	0.644	0.664
GradientBoostTree	F_1	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	Jaccard	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
libSVM	F_1	0.997	0.978	0.994	0.972	0.994	0.986	0.997	0.977	0.948	0.943
	Jaccard	0.994	0.957	0.989	0.946	0.989	0.973	0.995	0.956	0.901	0.892
libLinear	F_1	0.992	0.914	0.983	0.942	0.964	0.941	0.983	0.954	0.890	0.911
	Jaccard	0.983	0.842	0.966	0.889	0.930	0.889	0.967	0.912	0.801	0.837
Knearest	F_1	1.000	0.965	0.994	0.978	0.981	0.986	1.000	0.989	0.956	0.949
	Jaccard	1.000	0.933	0.989	0.957	0.962	0.973	1.000	0.978	0.916	0.904
Multi-classLogistic	F_1	0.977	0.957	0.963	0.934	0.961	0.937	0.978	0.946	0.892	0.885
	Jaccard	0.956	0.918	0.929	0.877	0.925	0.881	0.957	0.897	0.805	0.794
MultiLayerPerceptron	F_1	0.986	0.948	0.938	0.925	0.951	0.957	0.962	0.949	0.913	0.930
	Jaccard	0.973	0.902	0.884	0.861	0.906	0.918	0.926	0.902	0.840	0.869
NaiveBayesianNet	F_1	0.942	0.914	0.940	0.891	0.011	0.967	0.949	0.762	0.793	0.883
	Jaccard	0.890	0.842	0.886	0.804	0.006	0.936	0.902	0.616	0.656	0.791
CRF											

Table 14: Inter-class accuracy on optdigits data set.

indicated that the overall accuracy was boosted by the accuracy of the class 1. Distribution trends of Jaccard coefficient was in accordance with that of F_1. This meant that the higher F_1 was, the higher Jaccard coefficient was in class 1. Further, support vector machines, random forests and CRF also had high F_1 and Jaccard coefficient in class 1.

From table 12, on data set Ionosphere, class 0 represents the presence of the fact, class 1 represents no presence of the fact. Performance was higher in class 0 than that in class 1. Stochastic gradient boosting to achieve the highest value. Random forests, CRF and multi-class adaboost classifier also got good performance.

From table 14, on optdigits dataset, class 0 to class 9 represent ten digit of 0 to 9. All algorithms had relatively equal performance in each class. Stochastic gradient boosting, the non-linear support vector machines, random forests, Multilayer Perceptron, linear multi-class support vector machines and multi-class logistic classifier had high performance in each class.

Algorithm		Class0	Class1	Class2	Class3	Class4	Class5	Class6	Class7
DecisionTree	F_1	0.238	0.618	0.718	0.550	0.304	0.397	0.471	0.397
	Jaccard	0.135	0.447	0.561	0.380	0.179	0.247	0.308	0.248
RandomForest	F_1	0.848	0.654	0.852	0.327	0.491	0.571	0.618	0.688
	Jaccard	0.736	0.486	0.742	0.196	0.326	0.400	0.448	0.524
ExtraTrees	F_1	0.805	0.816	0.859	0.779	0.590	0.744	0.652	0.777
	Jaccard	0.673	0.689	0.752	0.638	0.419	0.593	0.483	0.635
BoostedClassifier	F_1	0.606	0.126	0.290	0.415	0.531	0.399	0.447	0.608
	Jaccard	0.435	0.067	0.170	0.262	0.362	0.249	0.287	0.437
GradientBoostTree	F_1	**1.000**	**1.000**	**1.000**	**1.000**	**0.997**	**1.000**	**1.000**	**1.000**
	Jaccard	**1.000**	**1.000**	**1.000**	**1.000**	**0.994**	**1.000**	**1.000**	**1.000**
libSVM	F_1	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
	Jaccard	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
libLinear	F_1	0.959	0.860	0.931	0.864	0.784	0.887	0.823	0.890
	Jaccard	0.921	0.754	0.871	0.761	0.645	0.797	0.699	0.803
Knearest	F_1	0.606	0.126	0.290	0.415	0.531	0.399	0.447	0.608
	Jaccard	0.435	0.067	0.170	0.262	0.362	0.249	0.287	0.437
Multi-classLogistic	F_1	0.941	0.801	0.873	0.820	0.736	0.866	0.770	0.871
	Jaccard	0.936	0.701	0.827	0.721	0.617	0.712	0.656	0.776
MultiLayerPerceptron	F_1	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
	Jaccard	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
NaiveBayesianNet	F_1	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
	Jaccard	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
CRF	F_1	0.835	0.647	0.843	0.318	0.485	0.565	0.546	0.659
	Jaccard	0.736	0.477	0.729	0.187	0.319	0.389	0.432	0.510

Table 15: Inter-class accuracy on Scene15 data set.

Algorithm		Class8	Class9	Class10	Class11	Class12	Class12	Class14
DecisionTree	F_1	0.463	0.330	0.142	0.234	0.220	0.326	0.280
	Jaccard	0.301	0.198	0.076	0.133	0.123	0.195	0.162
RandomForest	F_1	0.495	0.410	0.034	0.197	0.071	0.282	0.482
	Jaccard	0.328	0.258	0.017	0.109	0.037	0.164	0.318
ExtraTrees	F_1	0.681	0.645	0.339	0.286	0.409	0.496	0.531
	Jaccard	0.517	0.477	0.204	0.167	0.257	0.330	0.362
BoostedClassifier	F_1	0.611	0.447	0.342	0.320	0.325	0.356	0.485
	Jaccard	0.440	0.288	0.206	0.190	0.194	0.216	0.320
GradientBoostTree	F_1	**0.997**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	Jaccard	**0.994**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
BoostedClassifier	F_1	0.623	0.459	0.309	0.288	0.324	0.415	0.528
	Jaccard	0.452	0.298	0.183	0.168	0.193	0.262	0.359
libSVM	F_1	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
	Jaccard	0.000	0.000	0.000	0.000	0.000	0.000	0.000
libLinear	F_1	0.862	0.898	0.633	0.645	0.701	0.663	0.685
	Jaccard	0.758	0.815	0.463	0.476	0.539	0.495	0.521
Knearest	F_1	0.611	0.447	0.342	0.320	0.325	0.356	0.485
	Jaccard	0.440	0.288	0.206	0.190	0.194	0.216	0.320
Multi-classLogistic	F_1	0.847	0.898	0.721	0.613	0.761	0.721	0.655
	Jaccard	0.889	0.668	0.775	0.694	0.582	0.764	0.625
MultiLayerPerceptron	. F_1 .	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
	Jaccard	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
NaiveBayesianNet	F_1	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
	Jaccard	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
CRF	F_1	0.488	0.402	0.028	0.188	0.066	0.277	0.466
	Jaccard	0.313	0.248	0.016	0.100	0.036	0.155	0.309

Table 16: Inter-class accuracy on Scene15 data set (cont.)

Algorithm	Binary data sets			Multi-class data sets	
	SPECTF	Ionosphere	Spam	optdigits	Scene 15
DecisionTree	1	30.9	79.9	202.9	19.562(second)
RandomForest	93.9	326.9	2.819	8112	34(minute)
ExtraTrees	266.0	437.0	3.036	12541	1hour22 minute
BoostedClassifier	202.9	451.9	1761	2.697	477.873（second）
GradientBoostTree	437.0	656.0	6732	53807	8hour42 minute
libSVM	46.9	108.9	915184	3276	Invalid
libLinear	389.9	749.0	3.165	1006	2hour10 minute
Knearest	1	16.0	280.9	708.9	120.054（second）
Multi-classLogistic	30.9	46.9	377.9	2153	190.660（second）
MultiLayerPerceptron	**857.9**	**4306**	**33727**	**172847**	**Invalid**
NaiveBayesianNet	30.9	93.0	63.0	375.9	Invalid
CRF	168.9	235.2	1.522	6420	66 minute

Table 17: Running time of twelve algorithms on five data sets (unit: millisecond except scene 15 data set).

From table 15 and table 16, on Scene15, class 0 to class 14 represented fifteen classes. Multilayer perceptron and non-support vector machines were failed because of computation cost, and naive Bayesian classifier was failed due to the huge storage. Stochastic gradient boosting, linear support vector machines achieved good performance, following by multi-class logistic classifier.

5.3 Running time performance on five data sets

From table 17, the running time of the 11 kinds by an algorithm on the five data sets can be seen that:

1. On a small data sets (SPECTE and Ionosphere), running time of multilayer perceptron was significantly slower than that of other algorithms, while other algorithms' running time were almost same. Linear support vector machines' (based on liblinear) running time was inversely lower than that of nonlinear support vector machines based on libsvm.

2. On the large data sets (spam, optidigits, scene15), the differences of running time were significant. It is clear that the linear support vector machines were significantly faster than the non-linear support vector machines.

3. For tree classifiers, decision tree was the fastest of all, following by random forests. The slowest was extremely randomized trees.

4. For boosting methods, stochastic gradient boosting was slower than the multiclass adaboost.

5. Due to the large dimensionality of data, non-linear support vector machines, and Bayesian multi-layer perceptron did not succeed.

6. CRF running time is better than ExtraTrees, but slower than RandomForest.

7. In short, for running time efficiency, naive Bayes classifiers, K nearest neighbor and decision tree were basically fast, following by random forests, multi-class logic and linear support vector machines. The stochastic gradient boosting was the slowest of all.

6 Conclusion and future work

This article compares 12 kinds of commonly used multi-classification algorithm. In the experiments, we found that:

1. The same algorithm on different data sets showed different performance. It was the key to choose a more adaptive algorithm based on the data set.
2. Stochastic gradient boosting achieved the best classification accuracy in all test data sets, but its running time was slower than other algorithms except the multilayer perceptron.
3. The composite classifiers performed well than single classifier. For example, stochastic gradient boosting, random forest, extremely randomized trees were all better than the basic decision tree. However at the same time, the more complex combination model was, the longer running time was.
4. Linear support vector machines achieved good results of both accuracy and total execution on large data sets while compared with the nonlinear support vector machine.

There are still some deficiencies in our comparative study, further research is need:

1. We compare only the basic algorithm of 12 kinds of algorithms, every algorithm has its variants, which are better than the original algorithms.
2. How to choose the optimal parameter settings for algorithm is critical for its performance. There are still more works need to be done.
3. When we deal with large data sets, how much sample should choose for training? How to find the best balance between training time and accuracy is worthy of further exploration.
4. The combination of classifiers can often lead to higher accuracy, but as mentioned above, model training time will significantly increase. Stochastic gradient boosting were obtained good accuracy in our tests on five data sets, however the running time is longer. Actually stochastic gradient boosting have many sub routine which has sub-iteration, so this will elapse many running time. Because the main routine are highly correlate the sub-iteration, so it cannot directly parallel the sub-iteration.

How to improve the running time performance with a little bit of decrease in accuracy is a meaningful research, in other words, we need to find a balance between accuracy and running time performance.

Acknowledgement

This work was supported in part by Scientific Research

Fund of Sichuan Provincial Education Department of China (No. 15TD0038) and Fundamental Research Funds for the Central University.

References

[1] ZHU J, ROSSET S, ZOU H, et al. Multi-class adaboost [J]. Ann Arbor, 2006, 1001(48109): 1612.

[2] SELFRIDGE O G. Pandemonium: a paradigm for learning in mechanization of thought processes [J]. 1958,

[3] KANAL L. Patterns in pattern recognition: 1968-1974 [J]. Information Theory, IEEE Transactions on, 1974, 20(6): 697-722.

[4] MINSKY M L. Logical versus analogical or symbolic versus connectionist or neat versus scruffy [J]. AI magazine, 1991, 12(2): 34.

[5] ROSENBLATT F: DTIC Document, 1961.

[6] BREIMAN L. Random forests [J]. Machine learning, 2001, 45(1): 5-32.

[7] FRIEDMAN J, HASTIE T, TIBSHIRANI R. Additive logistic regression: a statistical view of boosting (With discussion and a rejoinder by the authors) [J]. The annals of statistics, 2000, 28(2): 337-407.

[8] FRENCH S. Group consensus probability distributions: A critical survey [J]. Bayesian statistics, 1985, 2(183-202.

[9] BENEDIKTSSON J A, SWAIN P H. Consensus theoretic classification methods [J]. Systems, Man and Cybernetics, IEEE Transactions on, 1992, 22(4): 688-704.

[10] BERNARDO J M, SMITH A F. Bayesian theory [J]. Measurement Science and Technology, 2001, 12(2): 221.

[11] FAN R E, CHANG K W, HSIEH C J, et al. LIBLINEAR: A library for large linear classification [J]. The Journal of Machine Learning Research, 2008, 9(1871-4.

[12] DUIN R, TAX D. Experiments with classifier combining rules [J]. Multiple classifier systems, 2000, 16-29.

[13] ALY M. Survey on multi-class classification methods [J]. Neural networks, 2005, 1-9.

[14] NILLSON N. Learning machines: Foundations of trainable pattern classifying systems [M]. McGraw-Hill, New York. 1965.

[15] KING R D, FENG C, SUTHERLAND A. Statlog: comparison of classification algorithms on large real-world problems [J]. Applied Artificial Intelligence an International Journal, 1995, 9(3): 289-333.

[16] BAUER E, KOHAVI R. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants [J]. Machine learning, 1998, 36(1): 2.

[17] LECUN Y, JACKEL L, BOTTOU L, et al. Comparison of learning algorithms for handwritten digit recognition; proceedings of the International conference on artificial neural networks, F, 1995 [C].

[18] DING C H, DUBCHAK I. Multi-class protein fold recognition using support vector machines and neural networks [J]. Bioinformatics, 2001, 17(2): 107-38.

[19] LI T, ZHANG C, OGIHARA M. A comparative study of feature selection and multi-class classification methods for tissue classification based on gene expression [J]. Bioinformatics, 2004, 20(15): 2429-37.

[20] FOODY G M, MATHUR A. A relative evaluation of multi-class image classification by support vector machines [J]. Geoscience and Remote Sensing, IEEE Transactions on, 2004, 42(6): 1335-43.

[21] HSU C-W, LIN C-J. A comparison of methods for multi-class support vector machines [J]. Neural Networks, IEEE Transactions on, 2002, 13(2): 415-25.

[22] CARUANA R, NICULESCU-MIZIL A. An empirical comparison of supervised learning algorithms; proceedings of the Proceedings of the 23rd international conference on Machine learning, F, 2006 [C]. ACM.

[23] KRUSIENSKI D J, SELLERS E W, CABESTAING F, et al. A comparison of classification techniques for the P300 Speller [J]. Journal of neural engineering, 2006, 3(4): 299.

[24] WITTEN I H, FRANK E. Data Mining: Practical machine learning tools and techniques [M]. Morgan Kaufmann, 2005.

[25] MURPHY K P. Machine Learning: a Probabilistic Perspective [J]. 2012,

[26] QUINLAN J R. Induction of decision trees [J]. Machine learning, 1986, 1(1): 81-106.

[27] BREIMAN L, FRIEDMAN J, STONE C J, et al. Classification and regression trees [M]. Chapman & Hall/CRC, 1984.

[28] HO T K. Random decision forests; proceedings of the Document Analysis and Recognition, 1995, Proceedings of the Third International Conference on, F, 1995 [C]. IEEE.

[29] BREIMAN L. Bagging predictors [J]. Machine learning, 1996, 24(2): 123-40.

[30] BREIMAN L. Statistical modeling: The two cultures (with comments and a rejoinder by the author) [J]. Statistical Science, 2001, 16(3): 199-231.

[31] IVERSON L R, PRASAD A M, MATTHEWS S N, et al. Estimating potential habitat for 134 eastern US tree species under six climate scenarios [J]. Forest Ecology and Management, 2008, 254(3): 390-406.

[32] GEURTS P, ERNST D, WEHENKEL L. Extremely randomized trees [J]. Machine learning, 2006, 63(1): 3-42.

[33] HASTIE T, TIBSHIRANI R, FRIEDMAN J, et al. The elements of statistical learning: data mining, inference and prediction [J]. The Mathematical Intelligencer, 2005, 27(2): 83-5.

[34] FREUND Y, SCHAPIRE R E. A decision-theoretic generalization of on-line learning and an application

to boosting [J]. Journal of computer and system sciences, 1997, 55(1): 119-39.

[35] FRIEDMAN J H. Stochastic gradient boosting [J]. Computational Statistics & Data Analysis, 2002, 38(4): 367-78.

[36] BURGES C J C. A tutorial on support vector machines for pattern recognition [J]. Data mining and knowledge discovery, 1998, 2(2): 121-67.

[37] CORTES C, VAPNIK V. Support-vector networks [J]. Machine learning, 1995, 20(3): 273-97.

[38] KEERTHI S S, SUNDARARAJAN S, CHANG K-W, et al. A sequential dual method for large scale multi-class linear SVMs; proceedings of the Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, F, 2008 [C]. ACM.

[39] CRAMMER K, SINGER Y. On the algorithmic implementation of multi-class kernel-based vector machines [J]. The Journal of Machine Learning Research, 2002, 2(265-92.

[40] COVER T, HART P. Nearest neighbor pattern classification [J]. Information Theory, IEEE Transactions on, 1967, 13(1): 21-7.

[41] LIU D C, NOCEDAL J. On the limited memory BFGS method for large scale optimization [J]. Mathematical programming, 1989, 45(1): 503-28.

[42] RONSENBLATT F. The perceptron: a probabilistic model for information storage and organization in the brain [J]. Psychological review, 1958, 65(386-408.

[43] BISHOP C M. Pattern recognition and machine learning [M]. Springer New York, 2006.

[44] WASSERMAN P D, SCHWARTZ T. Neural networks. II. What are they and why is everybody so interested in them now? [J]. IEEE Expert, 1988, 3(1): 10-5.

[45] Hang Li. Statistic Learning [M]. Tsinghua press, 2012.

[46] COLLOBERT R, BENGIO S. Links between Perceptron, MLPs and SVMs; proceedings of the Proceedings of the twenty-first international conference on Machine learning, F, 2004 [C]. ACM.

[47] GOULD S. DARWIN: A Framework for Machine Learning and Computer Vision Research and Development [J]. Journal of Machine Learning Research, 2012, 13(12): 3499-503.

[48] CHANG C C, LIN C J. LIBSVM: a library for support vector machines [J]. ACM Transactions on Intelligent Systems and Technology (TIST), 2011, 2(3): 27.

[49] FAWCETT T. An introduction to ROC analysis [J]. Pattern recognition letters, 2006, 27(8): 861-74.

[50] DAVIS J, GOADRICH M. The relationship between Precision-Recall and ROC curves; proceedings of the Proceedings of the 23rd international conference on Machine learning, F, 2006 [C]. ACM.

[51] JACCARD P. The distribution of the flora in the alpine zone. 1 [J]. New Phytologist, 2006, 11(2): 37-50.

[52] ASUNCION A, NEWMAN D J. UCI machine learning repository [M]. 2007.

[53] KURGAN L A, CIOS K J, TADEUSIEWICZ R, et al. Knowledge discovery approach to automated cardiac SPECT diagnosis [J]. Artificial Intelligence in Medicine, 2001, 23(2): 149-69.

[54] LAZEBNIK S, SCHMID C, PONCE J. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories; proceedings of the Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, F, 2006 [C]. IEEE.

[55] VEDALDI A, ZISSERMAN A. Efficient additive kernels via explicit feature maps [J]. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2012, 34(3): 480-92.

[56] Bradski G, Kaehler A (2008) Learning OpenCV: Computer vision with the OpenCV library. O'Reilly Media, Incorporated

[57] Delashmit WH, Manry MT (2005) Recent developments in multilayer perceptron neural networks. In: Proceedings of the seventh Annual Memphis Area Engineering and Science Conference, MAESC.